Proceedings of the 2002 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
EPFL, Lausanne, Switzerland • October 2002

Autonomous Coverage Operations In Semi-Structured Outdoor Environments

Parag H. Batavia, Stephan A. Roth, Sanjiv Singh

Robotics Institute
Carnegie Mellon University
Pittsburgh, USA
{parag, stephan+, ssingh}@ri.cmu.edu

Abstract

This paper presents a comprehensive navigation system capable of extended coverage operations in semi-structured environments. By semi-structured, we mean pre-mapped terrain that although locally smooth has potentially large changes in elevation and that is generally free of obstacles. The system has three key capabilities -- it is able to track specified paths with high accuracy, detect small obstacles reliably, and plan coverage patterns to completely cover a specified area. These technologies have been combined and implemented on a mobile robot, which has accumulated over 90km of autonomous operation to date. Here we report on the components, the architecture, and experimental results.

1. Introduction

This paper presents a comprehensive autonomous system for full coverage and traversal of semi-structured outdoor terrain. By semi-structured, we mean smooth terrain smooth with potentially large elevation changes, and with a low density of (potentially small) obstacles. Examples of this type of terrain include parks, sports fields, and golf courses. The major challenges for developing such a system are: (a) high accuracy tracking and localization, to allow for operation near potentially dangerous areas, (b) robust obstacle detection, which can operate in terrain with changing elevation, while still detecting small (15cm) obstacles, and (c) complete coverage planning to automatically cover a specified area completely. Since our intended application is commercial, ease of use and low cost are also important. This paper presents a system that exhibits high accuracy, high repeatability, robust obstacle detection, and complete coverage. This system has been in operation for months and has accumulated over 90km of autonomous operations, low cost, and simple to use; it has been tested and operated by non-technical people.

Our work has applications to automated mowing, spraying fertilizer, applying paint to sports fields, and examining large areas of grass for damage or disease. These types of operations require semi-skilled labor, which can be either expensive or in short supply. Furthermore, there can be labor-imposed constraints that specify when these operations can be performed. An autonomous system which requires minimal supervision could operate at night, when labor is not available.

There has been a great deal attention to parts of the problem of autonomous operation in semi-structured environments. For example, research in robust localization has been used to automate machines such as straddle carriers for transport of large containers in a port setting [8]. Research in tracking of paths for automated underground mining machines is presented in [19]. Automation of agricultural machines has concentrated mainly on navigation and tracking of local features [4][10][16][17], and has typically not focused on obstacle detection or coverage. A recent system has automated a tractor [21] capable of path tracking and localization. The system is able to detect obstacles based on texture and color segmentation, but there is no automated coverage generation. Other systems have developed random coverage patterns for covering small areas [9][11]. In comparison, our system is capable of tracking, localization, obstacle detection and coverage. Also unique with our system is the requirement to track specified paths to high accuracy and reliably detect very small obstacles.

Our approach to tracking and coverage uses paths based on an absolute reference frame. Although geo-referenced coverage patterns can potentially be generated using digital maps, we use a teach-playback system in which an operator drives the outline of a desired coverage area as input to a coverage generation algorithm. This makes training and operation easy for the user, and minimizes the work load.

Obstacle detection is also a major component of autonomous systems. Stereo [3][15][24], radar [14], and laser range finders [18] have all been used in the past for obstacle detection. Previous approaches based on stereo and other forms of passive vision such as color segmentation [1][12][23] suffer from lighting, color constancy, and dynamic range effects which cause false positives and false negatives. Radar is good for large obstacles, but localization is an issue due to wide beam widths. Single axis laser scanners only provide information in one direction, and can be confounded by unmeasured pitching motion and mis-registration. Two axis scanners are also used, which provide more information, but are very costly.

In recent work, we have developed a custom obstacle detection system [2]. Our approach is to mechanically pan a single axis scanner to provide horizontal and vertical coverage. We also have developed a novel obstacle detection algorithm capable of detecting small (15 cm) discrete

obstacles in curving but locally smooth terrain. We further assume that all areas are equally traversable, except for the presence of discrete obstacles.

Section 2 presents details about the core path tracking, navigation, obstacle detection, user interface modules, and overall architecture. Section 3 presents results of individual components, along the results of three months of outdoor testing on a mobile platform, in which time the robot autonomously covered 82,000m^2 of turf.

2. Subsystems

Here we present methods which enable complete autonomous operation in semi-structured terrain. These include cm-level path tracking, cm-level localization, robust obstacle detection in curving terrain, and an intuitive user interface which is capable of being used with minimal training.

2.1. Path Tracking

We use *pure-pursuit* [7], a well understood, intuitive path tracker, to achieve tracking accuracy on the order of +/- 5cm, when measured against the localization system. This accuracy is measured with reference to a path, which is represented as a series of points in the global coordinate frame. The path tracking error at any instant in time is the straight-line distance from the vehicle to the closest point on the path.

Pure pursuit operates by searching for a point along the path that is a *look-ahead* distance away from the control point. The steering angle is proportional to the y coordinate of the look ahead point when translated to the vehicle's reference frame, as shown in Figure 1. The basic form of pure-pursuit is a proportional controller. We have added an integral term to this to reduce bias error. Equation (1) gives the relationship between the look-ahead point and the resulting steering curvature. K_p is a proportional gain and K_i an integral gain.

$$\gamma = K_p(-e_x\sin\Theta + e_y\cos\Theta) + K_i\int(-e_x\sin\Theta + e_y\cos\Theta)dt \qquad (1)$$
$$e_x = x_f - x_g \qquad e_y = y_f - y_g$$

We have also modified the path tracker to compensate for constant delays in the vehicle's control system by using a kinematic model of the vehicle to estimate future position.

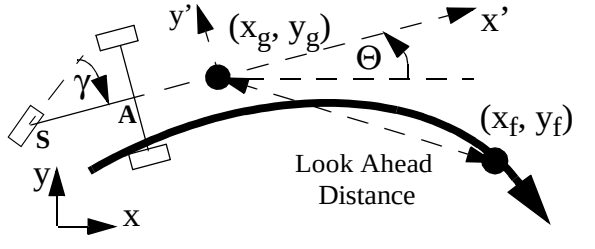

Figure 1: While following a path, the pure pursuit algorithm locates a point along the path at the a look ahead distance away from the control point. The steering angle is proportional to the look ahead point's y' error.

This position is used as input to the pure pursuit algorithm. As shown in Section 3.1, the maximum error is 5 cm on straight sections, and 10 cm on 2.5m radius curves.

2.2. Localization

Our intended applications require localization accuracy of 5cm over large areas. While the terrain is assumed to be smooth it can be populated with trees, buildings and other large obstructions. When line of sight to GPS satellites is obstructed, such as in areas of dense tree cover, localization is degraded, and is sometimes unavailable. In practice we have to augment our DGPS (2 cm accuracy) with wheel encoders and a fiber optic gyro to provide dead-reckoning.

The chosen sensors have different update rates and different error characteristics. We therefore use an extended Kalman filter formulation, described in [13], to fuse the sensor information. The state, $\bar{x}$ measured in the kalman filter is a vector containing the vehicles's x, y position, forward velocity, v, the yaw, Θ and yaw rate.

$$\bar{x} = \begin{bmatrix} x & y & v & \Theta & \dot{\Theta} \end{bmatrix}^T \qquad (2)$$

The kalman filter algorithm is broken into two steps, the time update (prediction) and the measurement update (correction.) During the time update, the kalman filter uses a simple dynamic model of the system to predict its new state.

$$\bar{x}^- = \bar{x} + dt\begin{bmatrix} v\cos(\Theta) & v\sin(\Theta) & 0 & \dot{\Theta} & 0 \end{bmatrix}^T \qquad (3)$$

Because the measurements are not correlated and the data are not synchronized, each sensor is handled individually. Occasionally, the GPS system returns invalid position data. To detect invalid measurement data, we analyze the filter residual to look for large discrepancies between the measured state and the predicted state. If this term is much larger than the measurement's reported standard deviation, then that measurement is discarded.

This combination of sensors and kalman filter provides a localization system that is very accurate in areas of unobstructed GPS positioning. The system is robust enough to drive through and recover from short periods (10-20 seconds) of degraded and absent GPS information, as we show in Section 3.2

2.3. Obstacle Detection

The obstacle detection subsystem is capable of detecting small (15cm) obstacles on curving terrain while moving at up to 2m/s, with a very low false positive rate. Although we initially relied solely on passive vision techniques, we have since adopted an active sensing system, making use of a custom-designed two-axis laser scanner. The motivation for this evolution is described in [2].

Figure 2: Two-axis laser scanner used for obstacle detection.

2.3.1. Hardware Design

Our current method mechanically sweeps a vertically-mounted single-axis laser scanner to produce a two-axis scanning system. Figure 2 shows a prototype of the system. The laser scanner generates a set of range points, called a *scan*, in a vertical direction, from top to bottom. The laser is mechanically rotated back and forth to produce a set of scans, which we refer to as a *sweep*. The rate and range of the sweep is determined by sensor resolution, vehicle speed, and desired update rate. [2] presents a trade-off analysis of the design parameters. The current system has the following characteristics:

- 100 degree (vertical) by 40 degree (horizontal) FOV
- 1 degree resolution in both directions
- 2Hz frame rate

Note that these parameters are mostly customizable. For instance, the horizontal resolution can be increased by decreasing the sweeping speed, at the cost of decreased frame rate or decreased FOV. This flexibility allows for focus of attention algorithms to be easily implemented.

The main advantage of this approach over commercial two-axis laser systems is cost. Commercial systems provide greater resolution and range than our system, but at a much higher (5-10x) cost.

2.3.2. Software Algorithm

The obstacle detection algorithm consists of three stages: classification, fusing, and filtering. As scan lines come in, each range point in the scan is classified as ‘obstacle’ or ‘freespace’. Scans are accumulated, and then obstacle pixels are clustered using a nearest-neighbor criterion, and candidate obstacles are then filtered based on statistics such as mass and size. Figure 3 illustrates this procedure.

The novelty of this algorithm is that point clustering and scan fusing is done in such a way as to be robust to data mis-registration, and hence, false positives.

The first stage, classification, is gradient-based. Gradient algorithms, as the name suggests, convert laser range data into cartesian coordinates in a vehicle-centric frame, and calculate the terrain gradient and look for areas which are not traversable based on slope. Our gradient calculation and classification method is similar to the method proposed by Chang [5]. Their algorithm processes each scan as it is accumulated, and classifies returns as obstacle or freespace based on gradients and heuristics. In our approach, as each scan is accumulated, it is registered to a fixed coordinate frame to account for platform motion. The gradient is computed along the dimension of the scan line. This is done in real time as each scan is received. A threshold is applied to the gradient to classify each pixel as ‘obstacle’ or ‘freespace’. Each classified scan is then added to a circular buffer which contains a time-history of scans. The duration of this ‘time-window’ determines the amount of data which will be fused when clustering.

At regular (user specified) intervals, the points classified as obstacles are clustered using a nearest-neighbor approach. The nearest-neighbor approach is in contrast to the conventional approach of registering all the data to a common frame, and *then* classifying pixels, which can lead to false positives due to mis-registration of data. Our approach avoids this by clustering obstacle pixels *after* they are classified. Therefore, small mis-registrations in the data lead to small errors in gross obstacle statistics, rather than errors in deciding whether an obstacle is present. The latter is relatively minor, while the former can lead to undesired stopping or obstacle-avoidance behavior. This is similar in motivation to previous work which merges traversability maps created from instantaneous range images rather than merging elevation maps because of indeterminacy of vehicle motion in between images [20].

There are two parameters which control the behavior of the algorithm: the gradient threshold, and the time window. The output rate is another parameter, but does not affect results. It is a function of available computing power-- a faster CPU can support a faster output rate than a slower

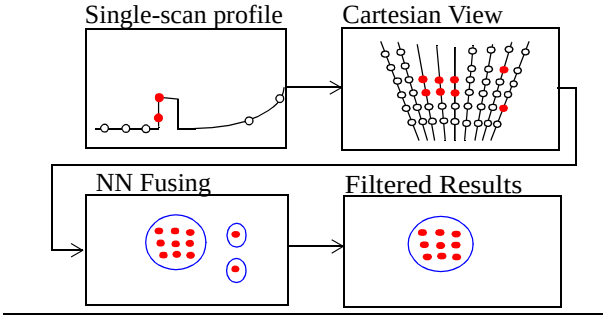

Figure 3: Obstacle Detection Algorithm overview. Each scan is classified as “obstacle” or “freespace”. Obstacle pixels from groups of scans are clustered using nearest-neighbor to generate candidate obstacles. Candidate obstacles are filtered based on gross statistics.

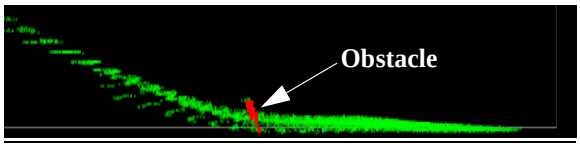

Figure 4: Profile view of a 22cm (9”) obstacle at the foot of a 20 degree slope.

CPU. Figure 4 shows a representative point cloud of an obstacle at the foot of a 20 degree slope. The obstacle is detected and properly classified, in the presence of registration errors, as evident in the side view of the slope.

2.4. User Interface

The user interface is a crucial component, since it is expected that non-experts will be using this system. We have adopted a “wizard” representation, which asks the operator a series of questions related to the task, and then sets up the appropriate options and begins operation. This representation can be limiting, as it forces the user into a linear, structured interaction with the system [6]. However, if properly designed, it can reduce the amount of training required to operate. Figure 5 shows a flowchart of the “wizard” interface. This shows the queries the user is presented, and the actions which are taken based on his responses. We also have an “expert” interface, which is suitable for use by people familiar with the details of operation. This interface is capable of being displayed directly on the vehicle, or remotely, over wireless ethernet (using either a laptop or a handheld computer as the client). This allows for remote monitoring and telemetry functions.

An initial qualitative test of the wizard interface has shown that it is easy to learn. Three untrained individuals were asked to use our testbed to construct a path by driving the outline of an area, and then activate autonomous operation. They were able to do so with little difficulty. While this initial result is positive, a more formal user study needs to be done to determine what improvements are needed to the interface.

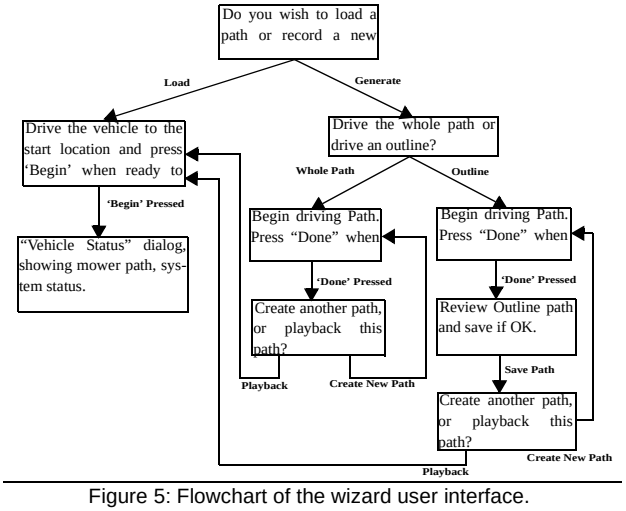

Figure 5: Flowchart of the wizard user interface.

2.5. System Architecture

The above sections presented components of an autonomous vehicle. This section describes how they are combined and interact with each other. Figure 6 shows the hardware and software architecture. The hardware architecture consists of sensing, actuation, and processing. The core sensing elements are the custom laser range finder for obstacle detection, and a GPS, gyro, and encoders for localization. The software architecture consists of multiple processes, each running on a single CPU, and communicating via standard IPC techniques such as semaphores, message queues, and shared memory. Cost and complexity were motivating factors in deciding to use a single CPU vs. multiple CPUs. Each process is either a publisher or subscriber of high level information such as obstacle lists, throttle/steering commands, or localization information. Safety monitoring is distributed, and implemented in each module. Therefore, the system gracefully degrades if individual modules or hardware components fail.

3. Results

We present results of each of the major subsystems, along with results from extended duration trials conducted on a mobile robot.

3.1. Tracking Results

Our application requires the vehicle to travel over relatively flat terrain with 5cm tracking accuracy over straight sections of path and 10cm over curved sections of path. To test the tracker’s performance we analyzed the tracking

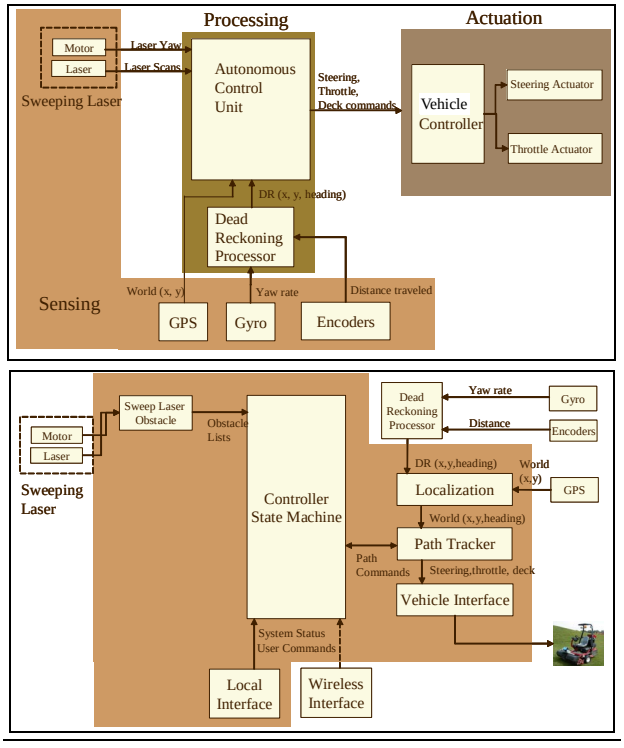

Figure 6: Hardware (top) and software (bottom) architecture

error over an 8km path covering 10,600m^2, at an average speed of 1.5m/s. The terrain was mostly flat with a few small hills.

To gauge the tracker's overall performance, we measured the distribution of the tracking error over the entire path and also over the path's straight sections. Overall, the standard deviation of the tracking error is 3.7cm., with 97% of the tracking error falling below 10 cm. Along the path's straight sections the standard deviation is 2.2cm, and 97% of the tracking error is less than 5cm. This level of tracking error meets our performance criteria for both straight-line and curved path tracking. Figure 7 shows a histogram of tracking error for an entire path (left) and for straight sections alone (right).

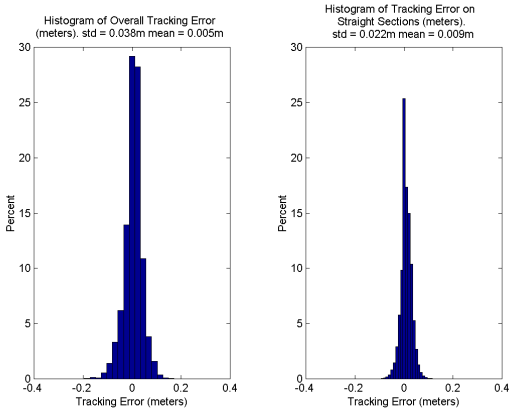

Figure 7: Histogram of tracking errors for entire path (left) and straight sections alone (right).

3.2. Localization Results

We evaluate the localization system's performance during GPS outages in two different tests. This first is by applying a simulated GPS dropout to localization data collected using our mobile platform while traveling at 1.5m/s, and the second is to subject the robot to real GPS outages by operating it underneath dense tree cover. The advantage of the first test is that it allows us to continue using GPS for ground truth measurements, so we can compare localization system estimates with actual position. The disadvantage is that the GPS dropout is artificially induced, and does not exhibit the same characteristics of a true dropout. I.e., a true dropout will usually result in degraded estimates, rather than no estimate at all. However, this test illustrates the response to a worst case scenario. Figure 8 shows a plot of localization error (top) and kalman filter standard deviation (bottom). The GPS outage begins at time T=40, and lasts for 10 seconds. During this time, the localization system is using dead-reckoning (based on gyro and encoders) alone to estimate position. Over the duration of the outage, the localization error, as measured against GPS position, never goes beyond 30cm. The kalman filter covariance increases to reflect its uncertainty.

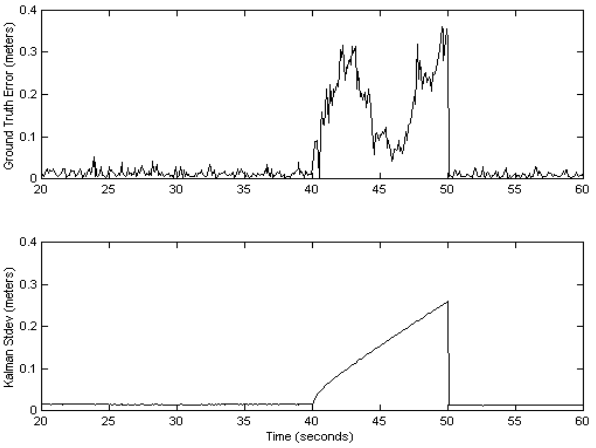

Figure 8: Performance of Kalman filter during simulated GPS outage. The top plot shows localization error, using GPS as ground truth. The bottom plot shows the kalman filter error estimate.

In the second test, we drove the vehicle in and out of an area of dense tree cover, which consistently caused a GPS outage. We then compared the kalman filter's calculated position with the vehicle's true position at a point 60m from the point the vehicle entered the trees. The deviation from the true position at this point was +/- 0.4m. The kalman filter's reported standard deviation was 0.5m, agreeing with the observed results. Given relatively short duration GPS outages, the kalman filter is able to maintain reliable localization.

3.3. Obstacle Detection Results

We have completed two tests of the obstacle detection subsystem. The first test involved measuring detection distances, while the second test involved an extended duration field trial, during which obstacles were placed in front of our robot testbed. During this trial, true positive, false positive, and false negative statistics were collected.

In the first test, we tested obstacle detection performance in two situations: flat terrain and curved terrain. Three different square obstacles were used, with sides of 15cm, 22cm, and 30cm. The tests were conducted on a moving platform driven by a person. An initial pass over obstacle-free terrain was used to adjust the gradient threshold such that no false positives were generated. When testing on hilly terrain, each obstacle was placed at the foot of a steep (20% grade) hill. We started 10m away from the obstacle and drove towards it at 1.5m/s, stopping at a distance of 0.5m from the obstacle. Table 1 shows results for combinations of obstacle size, terrain type, and time-window. The time-window parameter controls how much data is fused when detecting obstacles. The entries in the table show the distance to obstacle when it was *first* detected. I.e., this is the maximum distance at which we can expect to detect the obstacle. We see that in all cases, the obstacle is detected earlier on flat terrain than on hilly terrain. This is most likely due to our use of a steep gradient threshold, combined with low density sampling at far (> 3m) distances

Table 1: Max Distance (m) at which Obstacle is Reliably Detected for Flat (Hilly) Terrain Using Different Time Windows (s)

Obs Size / Window	15cm	22cm	30cm
0.5s	3.28 (2.13)	4.24 (4.00)	6.15 (7.01)
1.0s	3.15 (1.90)	4.38 (4.05)	7.45 (7.13)
1.5s	3.09 (1.83)	4.30 (4.09)	7.50 (7.07)
2.0s	3.25 (1.89)	4.24 (4.05)	7.73 (7.01)

due to the geometry of the laser. In all cases, the obstacles were detected within the stopping distance of the mobile platform we use, which travels at 1.5-2m/s. Interestingly, the time window has very little impact on the detection distance. This is because the robustness to false positives is great enough that the algorithm can reliably detect an obstacle with a small number of hits at extreme range. The additional hits later on do not add to the detectability, although they do add to the confidence. For instance, at 2.0m, a 15cm obstacle generates 9 hits with a time window of 0.5s. However, if the time window is 2.0s, then there are 32 hits.

During the obstacle detection field trial, objects between 5" and 12" in height were placed in front of the robot. A successful detection was defined as the robot stopping without impacting the object, regardless of final distance to the obstacle. This trial involved approximately 10km of autonomous travel, which resulted in 128 true positives, 10 false positives and 5 false negatives. Of the 10 false positives, 2 were due to large tufts of grass which exceeded the OD height threshold. Seven were due to dust temporarily obscuring the laser scanner (this laser is a first pulse laser scanner, which is sensitive to dust and fog), and one false positive was due to an unknown event. All five of the false negatives were in sharp turns, involving small (5-6") obstacle, where the inside of the turn wasn't properly visible. This has since been corrected by further biasing the laser sweeping angle during turns.

3.4. Extended Duration Operation Results

In addition to individually testing each subsystem, we have extensively tested the combined operation of the components on a mobile robot. The vehicle is a modified riding lawn mower, used to mow golf courses and sports fields, and is pictured in Figure 9. It has been retrofitted with all of the sensing, actuation, and software previously described. Note that there is nothing inherent about this vehicle that makes it the only possible test platform. The work described here could be used on a variety of different vehicles.

Figure 9: Mobile robot test platform is a retrofitted mower used for sports fields and golf courses.

Our testing has been in two main areas. The first is a test site roughly 4,000 m^2 in area, with a total distance traveled of approximately 5km for complete coverage. This site has changing elevation, with slopes up to 20 degrees. We have covered this area 13 times, although obstacle detection was not active in all 13 trials. The coverage pattern for this area is shown in Figure 10. The coverage pattern for convex

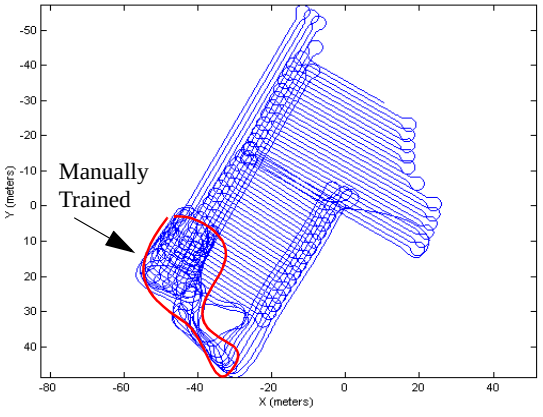

Figure 10: Coverage pattern of test area. This pattern was partially manually trained and partially automatically generated.

portions of this area were automatically generated, although a small area was manually taught. This portion is indicated in the figure. The second location was a 300-meter fairway, roughly 11,000 m^2 in size, with a total distance traveled of 8km for complete coverage. Table 2 shows accumulated statistics of the trials.

Human intervention was occasionally necessary, due to persistent obstacle detection false positives (two times), and hardware failures (one time). Aside from those interventions, the system autonomously covered the desired areas without interruption, and with no obstacle detection false negatives.

Table 2: Extended Duration Trial Statistics

	Number of Trials	Area Covered (m^2)	Distance Travelled (km)	Total Duration (hours)
Test Area	**13**	**52K**	**65**	**17**
Fairway	**2**	**22K**	**16**	**3**
OD Field Trial	**2**	**8K**	**10**	**2.5**
Total	**17**	**82K**	**91**	**22.5**

4. Conclusion and Future Work

In this paper, we have presented components for high accuracy path tracking, localization, and robust obstacle detection. These methods are intended for use on semi-structured outdoor terrain, and have been evaluated using a mobile robot with over 20 hours of unattended operation, covering a total area of approximately 82,000m^2. Future work will concentrate on improving localization to reduce the dependence on high accuracy (and high cost) GPS, and additional testing.

References

[1] P. H. Batavia., and S. Singh, "Obstacle Detection Using Adaptive Color Segmentation and Color Homography," Proceedings of the International Conference on Robotics and Automation. IEEE, May, 2001

[2] P. H. Batavia and S. Singh., "Obstacle Detection in Smooth, High-Curvature Terrain," Proceedings of the International Conference on Robotics and Automation Washington, D.C., 2002

[3] P. Bellutta et. al., "Terrain Perception for Demo III," Proceedings of the IEEE Intelligent Vehicles Symposium 2000, Dearborn, MI, 2000.

[4] J. Billingsley and M. Schoenfisch, "Vision Guidance of Agricultural Vehicles," Autonomous Robots, Vol 2. 1995

[5] T. Chang., et. al., "Concealment and Obstacle Detection for Autonomous Driving." Proceedings of the International Association of Science and Technology for Development - Robotics and Applications Conference, Santa Barbara, CA, 1999.

[6] A. Cooper, *About Face: The Essentials of User Interface Design*, Hungry Minds Inc., New York, 1995.

[7] R. Coulter, "Implementation of the Pure-Pursuit Path Tracking Algorithm," Carnegie Mellon University Technical Report CMU-RI-TR-92-01, 1992.

[8] H.F. Durrant-Whyte, "An Autonomous Guided Vehicle for Cargo Handling Applications," International Journal of Robotics Research, Vol 15, 1996.

[9] Friendly Robotics Robomower, on web at http://www.friendlyrobotics.com

[10] J.B. Gerrish et al., "Self-Steering Tractor Guided by Computer Vision," Applied Engineering in Agriculture, Vol 13, No 5, 1997

[11] Husqvarna Auto-Mower, on web at http://www.usa.husqvarna.com

[12] J. Hyams and M. Powell and R. Murphy, "Cooperative Navigation of Micro-rovers using Color Segmentation,' Journal of Autonomous Robots, Vol. 9, Num. 1, pp. 7-16, August, 2000.

[13] A Kelly, "A 3D State Space Formulation of a Navigation Kalman Filter for Autonomous Vehicles," Carnegie Mellon University Technical Report CMU-RI-TR-94-19, 1994

[14] D. Langer and T. Jochem, "Fusing Radar and Vision for Detecting, Classifying, and Avoiding Roadway Obstacles," Proceedings of the IEEE Symposium on Intelligent Vehicles, 1996.

[15] R. Mandelbaum et. al. "Real Time Stereo Processing, Obstacle Detection, and Terrain Estimation from Vehicle-Mounted Stereo Cameras," Proceedings of the Applications of Computer Vision, 1998.

[16] M. Ollis, and A. Stentz, "Vision-Based Perception for an Autonomous Harvester," Proceedings of the IEEE Conference on Intelligent Robot Systems, Sept., 1997.

[17] T. Pilarski et. al., "The Demeter System for Automated Harvesting," To appear in Autonomous Robots, July 2002.

[18] N. Roy, et al., "Towards Personal Service Robots for the Elderly," Proceedings of the Workshop on Interactive Robots and Entertainment, 2000.

[19] S. Scheding et. al., "An Experiment in Autonomous Navigation of an Underground Mining Vehicle," IEEE Transactions on Robotics and Automation, Vol. 15., No. 1., Feb. 1999.

[20] S. Singh et al., "Recent Progress in Local and Global Traversability for Planetary Rovers," Proceedings of the IEEE International Conference on Robotics and Automation, 2000, IEEE, April, 2000.

[21] A. Stentz et. al., "A System for Semi-Autonomous Tractor Operations," To appear in Autonomous Robots, July 2002.

[22] A. Stentz et. al., "A Robotic Excavator for Autonomous Truck Loading," Proceedings of International Conference on Intelligent Robots and Systems, 1998, Victoria, Canada.

[23] I. Ulrich and I. Nourbakhsh, "Appearance-Based Obstacle Detection," AAAI National Conference on Artificial Intelligence, Austin, TX, August 2000.

[24] C. Urmson and M.B. Dias, "Stereo Vision Based Navigation for Sun-Synchronous Exploration," Proceedings of the International Conference on Robotics and Automation, 2002, IEEE, May, 2002.

[25] A. Veatch and L.S. Davis., "Efficient Algorithms for Obstacle Detection Using Range Data," Computer Vision, Graphics, and Image Processing, 50(1), April 1990.

[26] Zhang, Y.L., S. A. Velinsky, and X. Feng. 1997. On the Tracking Control of Differentially Steered Wheeled Mobile Robots. Journal of Dynamic Systems, Measurement, and Control: 455-461

Acknowledgements

We would like to thank Adam Szymanski for his work on this project, along with members of the Autonomous Spraying team, in particular, Carl Wellington, who developed original versions of the pure-pursuit and localization code.